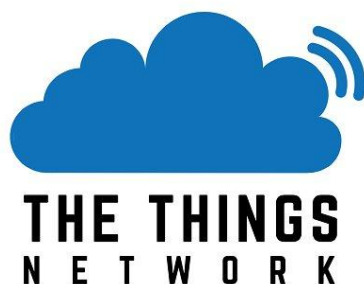


Terabee LoRa Level Monitoring XL

Integration with The Things Network (TTN) server



v1.0

Technical support: support@terabee.com
Sales and commercial support: terabee-sales@terabee.com

Table of contents

Introduction	3
Symbols explanation	3
Device integration with The Things Network	4
The Things Network (TTN) account	4
Creating a New Application	5
Creating a New Device	6
OTAA vs ABP as activation methods	11
Testing network connection	11
Receiving data from Terabee device to TTN server	12
Decoding uplink payload	14
Appendix A	16
General uplink payload structure	16
Error codes (byte 5) bit structure	16
Uplink payload decoding function	16

Introduction

The purpose of this document is to give guidelines for Terabee LoRa Level Monitoring XL device (further in text referred as Terabee device) integration with the The Things Network (TTN) server. This includes instructions on device registration and setup, receiving uplink frames as well as decoding the payload.

Symbols explanation

The following symbols are used within the document:



This symbol indicates important messages or specific recommendations in order to operate the product in the intended manner

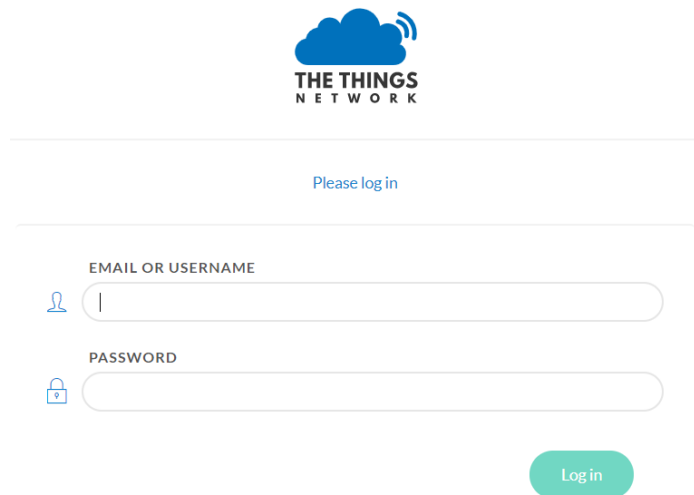
Device integration with The Things Network

The Things Network (TTN) account

In your browser, navigate to : <https://www.thethingsnetwork.org/>

For users with existing TTN accounts, select LOGIN on the home page, and enter corresponding credentials (username / password).

For users without a TTN account, select SIGN UP and follow the onscreen instructions for new account creation.



The screenshot shows the TTN login interface. At the top center is the TTN logo, which consists of a blue cloud with a signal icon and the text 'THE THINGS NETWORK' below it. Below the logo is a horizontal line, followed by the text 'Please log in'. Another horizontal line is below that. There are two input fields: the first is labeled 'EMAIL OR USERNAME' and has a person icon to its left; the second is labeled 'PASSWORD' and has a lock icon to its left. Below the password field is a green 'Login' button.

After successful registration, navigate to CONSOLE for further Terabee device registration.

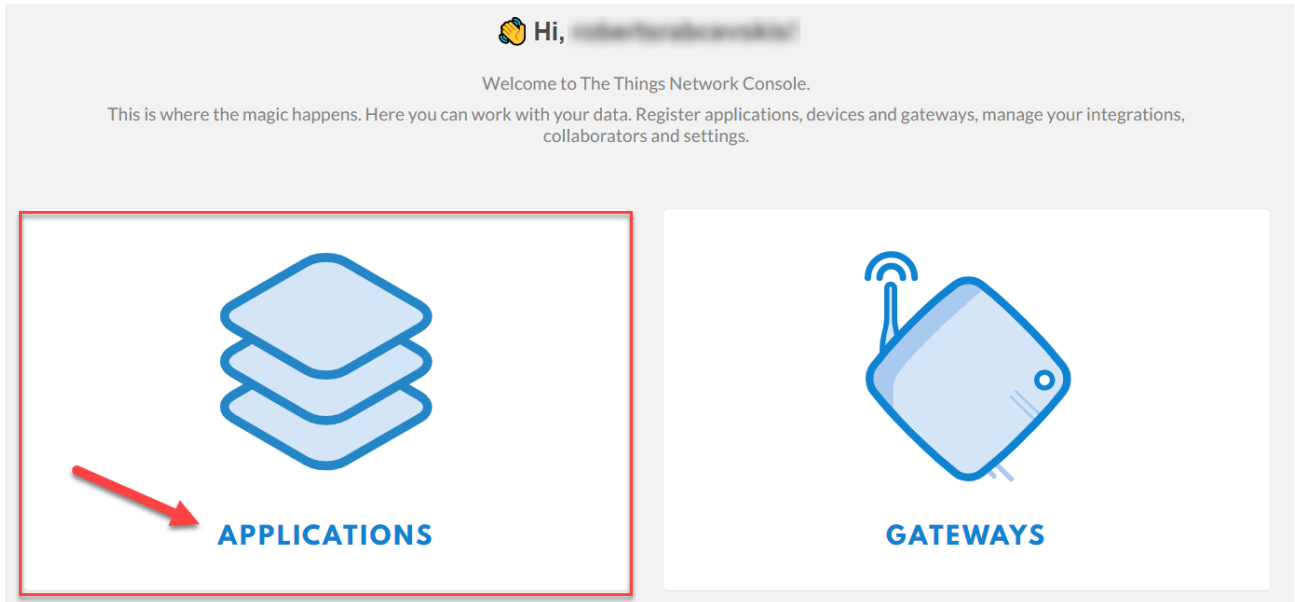


Please ensure that the Terabee device is situated in radio range) of at least one TTN LoRa gateway, corresponding to its provisioned installation spot on the field. To verify network coverage in your area, use the [TTN gateway map](#). If in doubt about coverage range, it is advised to use a LoRa network tester in order to evaluate TTN network coverage strength in a particular area

If no coverage is provided by TTN in a selected area, the user can install a private gateway and register it under the TTN community. Please consult the selected LoRa gateway manufacturer for further instructions.

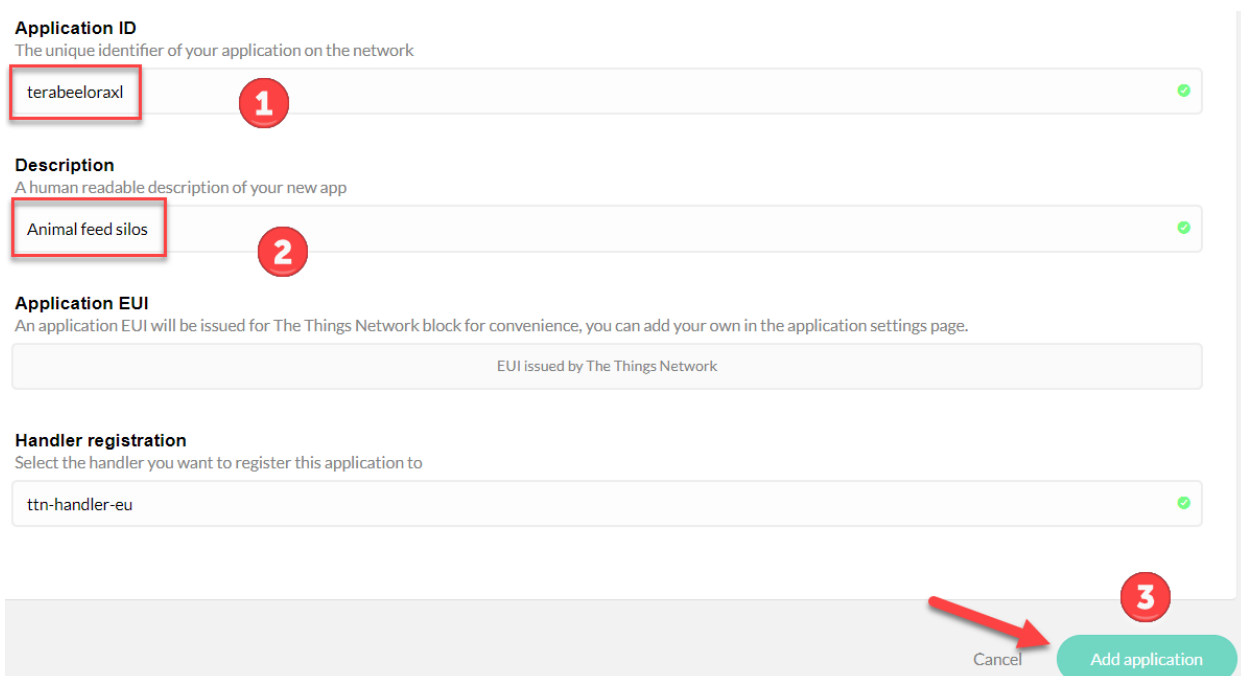
Creating a New Application

Under the CONSOLE section, select APPLICATIONS.



Then, select ADD APPLICATION (upper left side).

Fill the first two fields including Application ID and Description fields. Right after that, the Application EUI code will be auto-generated and available to the user in the corresponding field.

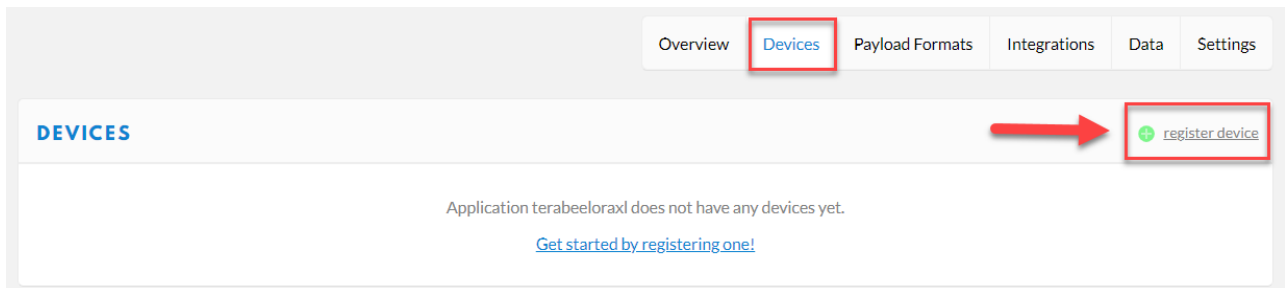
The screenshot shows the 'Add Application' form in the The Things Network Console. The form has four sections: 'Application ID' (The unique identifier of your application on the network) with the value 'terabeeloraxl' (marked with a red box and '1'); 'Description' (A human readable description of your new app) with the value 'Animal feed silos' (marked with a red box and '2'); 'Application EUI' (An application EUI will be issued for The Things Network block for convenience, you can add your own in the application settings page.) with the value 'EUI issued by The Things Network'; and 'Handler registration' (Select the handler you want to register this application to) with the value 'ttn-handler-eu'. At the bottom right, there is a red arrow pointing to the 'Add application' button (marked with a red box and '3') and a 'Cancel' button.

For the handler registration field, it is advised to select the geographically closest TTN server in the region. For Europe, this is *ttn-handler-eu* (default).

Once all fields are filled, click on ADD APPLICATION at the bottom right part of the page.

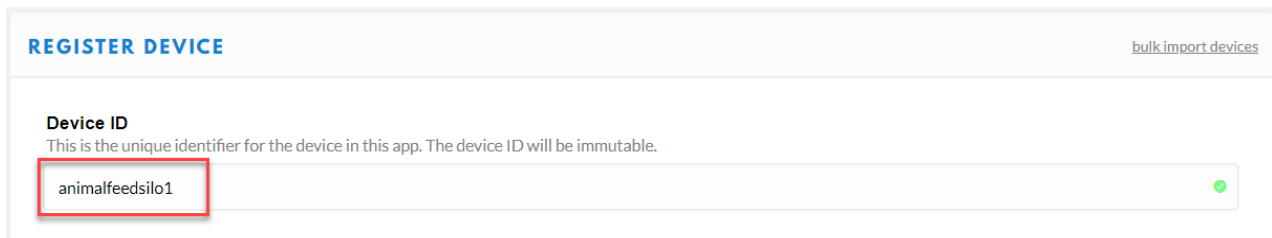
Creating a New Device

1. Once a New Application has been created, navigate to the DEVICES section. Click on REGISTER DEVICE at the top right corner of the dialog box.



A new area will open with 4 input fields : Device ID, Device EUI, App Key and App EUI.

2. Select and input a Device ID. It is a unique identifier for the specific Terabee device in the created application. Please note that once saved, the Device ID is fixed without the ability to modify it.



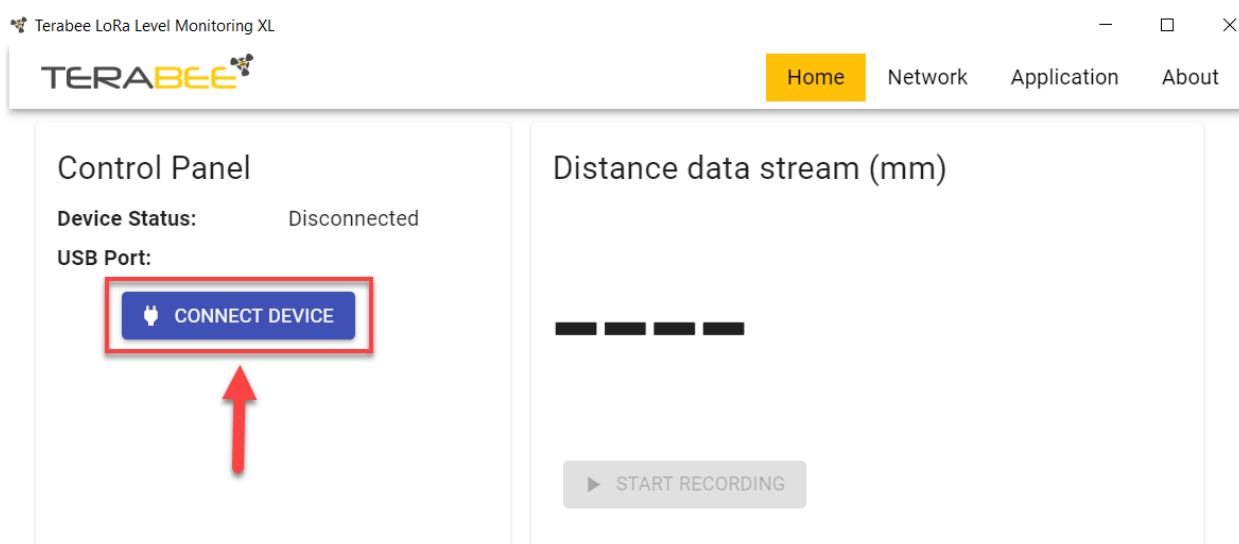
3. Next, complete the Device EUI and Application KEY fields (these can be modified at a later stage, after registration).

The user can either employ keys provided with the Terabee device, or use the auto-generated keys from the TTN platform. The following instructions show how to apply DevEUI and AppKEY provided by the Terabee device.

1. Connect the Terabee device to the PC using the available USB 2.0 Micro-B connection and cable.



II. Launch the Terabee Configuration GUI, and click on CONNECT DEVICE.



III. Navigate to the NETWORK tab, the DevEUI and AppKey codes are autofilled and available under the corresponding fields.

Activation Method:
 OTAA = Over The Air Activation
 ABP = Activation By Personalization

OTAA ▼

Device EUI:
 unique 64 bit end-device identifier

devEUI
 00bfa48881bae988

Application EUI:
 unique 32 bit application identifier

appEUI

Application KEY:
 128 bit identifier to secure communication between device and network

appKey
 81401dd3699ac1b0c8a17766f

COPY

IV. Copy the keys from the Configuration GUI into the TTN software, DevEUI and AppKey fields.

REGISTER DEVICE [bulk import devices](#)

Device ID
 This is the unique identifier for the device in this app. The device ID will be immutable.

animalfeedsilo1 2

Device EUI
 The device EUI is the unique identifier for this device on the network. You can change the EUI later.

00 BF A4 88 81 BA E9 88 3 8 bytes

App Key
 The App Key will be used to secure the communication between you device and the network.

81 40 1D D3 69 9A C1 B0 C8 A1 77 66 5E 20 02 C5 16 bytes

App EUI
 70 B3 D5 7E D0 03 FF C1

Cancel Register 4

As an alternative, DevEUI and AppKEY codes can also be extracted from a QR code, located on the printed label, inner side of the top screwable lid.

Scan for DevEUI, AppKEY

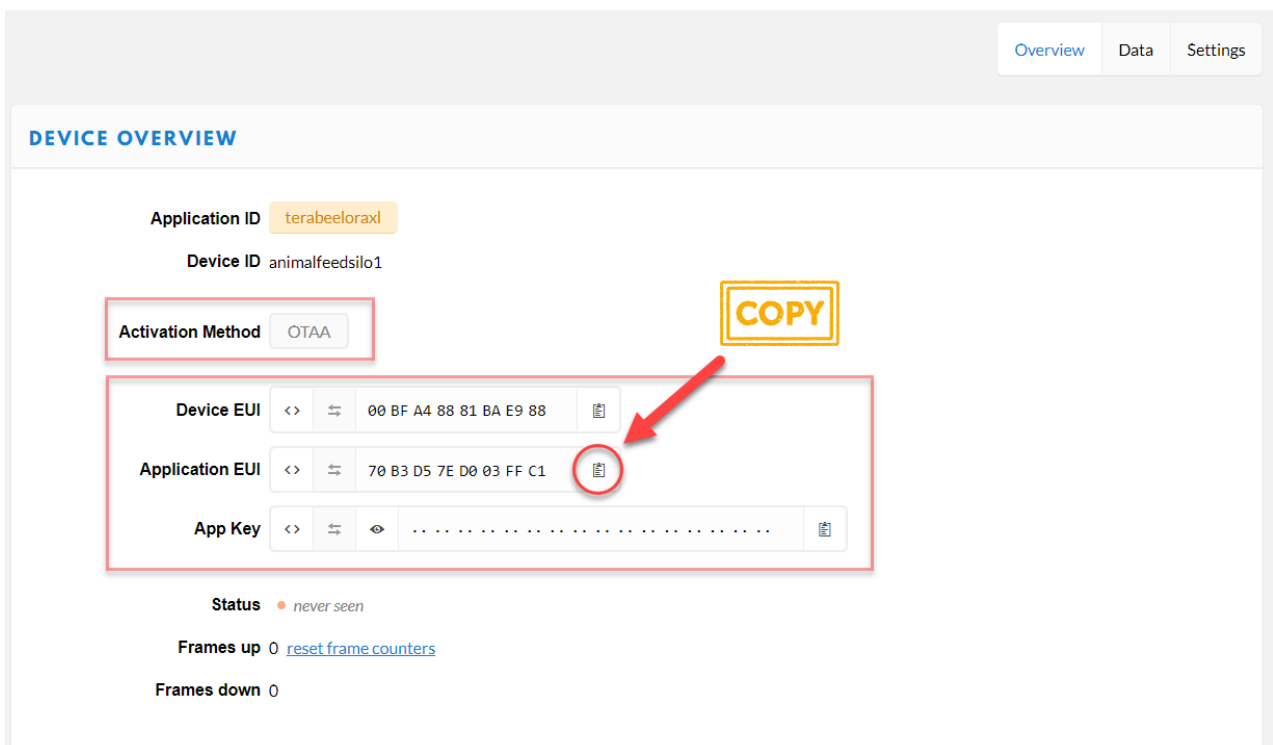


DevEUI: 001bc56701144dd
 AppKey: e73947e6b8f87d816ff31f154ff89950
 Serial N: 32165487

4. Click on REGISTER at the bottom left corner in the TTN platform.

The New Device is now registered on the TTN server with device information available under DEVICE OVERVIEW (top right).

It is suggested to maintain the Activation method to OTAA (Over the Air Activation) for higher data transmission security. If your application requires operation on ABP (Activation by Personalization), please make sure to select this accordingly in the Terabee Configuration GUI and the TTN platform.



5. Copy the auto-generated App EUI from the TTN software and paste it into the Terabee Configuration GUI → Network tab → Application EUI field. Click on SAVE TO DEVICE (bottom left of the Network tab window) to register the latest parameters to the device.


Activation Method:

OTAA = Over The Air Activation
ABP = Activation By Personalization

OTAA ▾

Device EUI:

unique 64 bit end-device identifier

devEUI
00bfa48881bae988 

Select Select All **Paste**

Application EUI:

unique 32 bit application identifier

appEUI
70B3D57ED003FFC1

Application KEY:

128 bit identifier to secure communication between device and network

appKey
81401dd3699ac1b0c8a17766!



The Application EUI is generated by the TTN software. If more than one application has been created, please select an appropriate one from the dropdown field.

6. Device registration on the TTN server is now completed.



In ABP mode, the Frameup counter MUST be reset every time the Device has been reset, otherwise for security reasons all Network messages will be dropped on the server side. Frame up check procedure can also be disabled during the configuration step in the TTN server, however this will reduce network security level.

In cases when the user prefers employing the auto-generated codes by the TTN software, please follow these simple steps (for OTAA method) :

- I. During the NEW DEVICE REGISTRATION step on TTN, select the "generate" function on the left side of the input field and after click on REGISTER (bottom right)
- II. Generated codes (DevEUI, AppKEY, AppEUI) are now available under the DEVICE OVERVIEW section. Use the "copy to clipboard" function for each of the codes.
- III. Assign the new codes to the Terabee Device using the Configuration GUI. Navigate to the NETWORK section, and paste the codes into the corresponding fields. Click on SAVE to DEVICE (bottom left of the Network tab) to register new parameters on the device.

OTAA vs ABP as activation methods

Over-the-Air Activation (OTAA) is the preferred and most secure way to connect with The Things Network. Devices perform a join-procedure with the network, during which a dynamic DevAddr is assigned and security keys are negotiated with the device.

In some cases you might need to hardcode the DevAddr as well as the security keys in the device. This means **Activating a device by Personalization (ABP)**. This strategy might seem simpler, because you skip the join procedure, but it has some downsides related to security.

Source : [TheThingsNetwork, Addressing & Activation](#)

Testing network connection

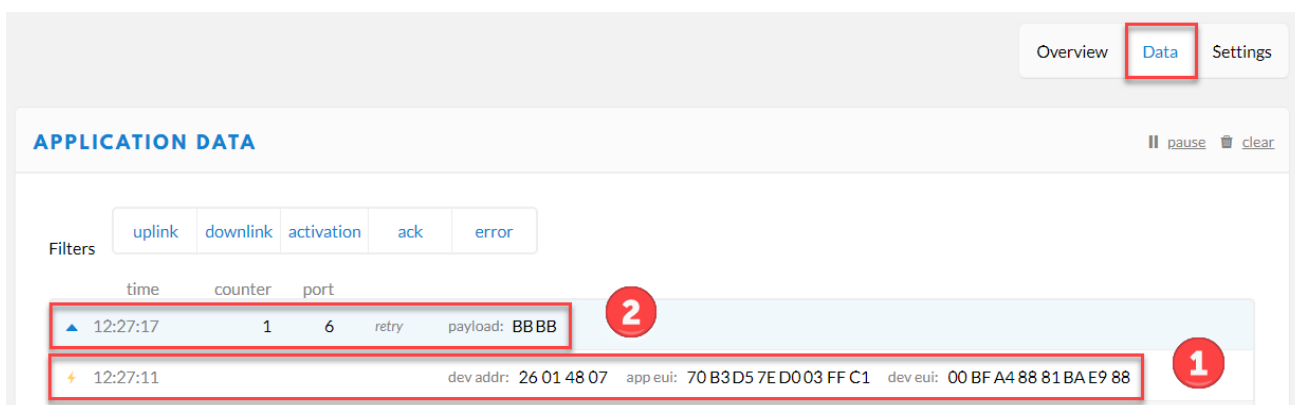
The Terabee Configuration GUI provides users with the possibility to test connection between the Terabee device and surrounding gateway/s.

1. In the GUI, under the Network tab, click on TEST NETWORK (bottom right).



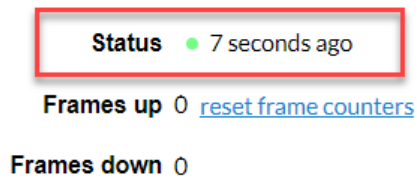
The Terabee device will initiate a join request procedure, exchanging unique identifiers with the network server. After a successful connection, an uplink message will be sent from the device with a test payload info **“BBBB”**. The device will then wait for an acknowledgement via downlink from the server.

The whole network test step can be monitored on the TTN platform, under the DATA field of the Device. An example of network join request & uplink frame is provided below.



time	counter	port	payload
12:27:11	1	6	dev addr: 26 01 48 07 app eui: 70 B3D5 7ED003 FF C1 dev eui: 00 BFA4 88 81BAE9 88
12:27:17	1	6	retry payload: BBBB

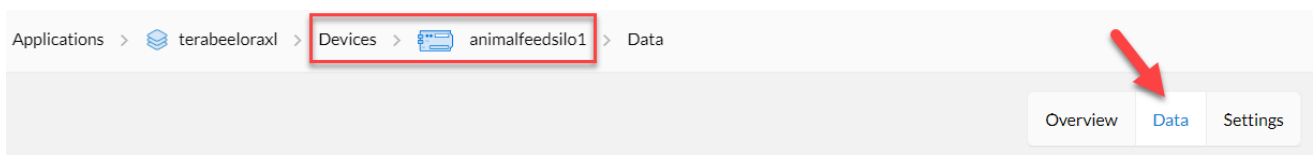
The Status indicator in the DEVICE OVERVIEW section will also change to green to signify a successful connection between the Terabee device and the TTN server.



“Frames up” indicates the number of uplink payloads received on the gateway from the device. “Frames down” indicates the number of downlink payloads sent from gateway to end device.

Receiving data from Terabee device to TTN server

1. Under the registered TTN device, navigate to the DATA tab (top right). This section provides users with uplink, downlink, activation and error data exchanged between device and the network.



Please make sure the Terabee device is disconnected from the PC or does not receive power supply from other devices via USB 2.0 interface, before activation.

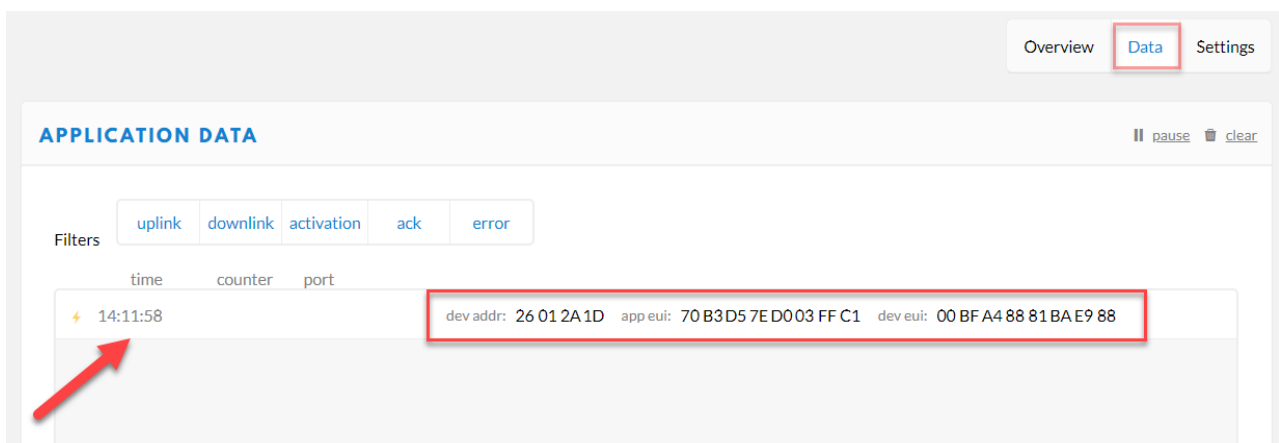
2. Activate the device by turning the onboard switch (under the device top lid) to (I) position. The Device will boot up and right after the network join request sequence will start. This is indicated by the onboard LED blinking constantly BLUE.



Before device activation, please ensure that the provided antenna is connected to the RP-SMA connector on the Terabee device.



A lightning bolt character will appear first in the DATA dialog window to indicate a successful exchange of unique keys (DevEUI, AppEUI, AppKEY) between the Terabee device and the TTN server. A device address is automatically appointed to the Terabee device by the network.



3. After the network join request has been accepted, the Terabee device will start transmitting data via LoRa protocol to the network server at regular intervals. Initial 15 transmissions will be sent every 2-3 minutes. Please note that the first uplink transmission can take up to 5-6 minutes.

Please consult the official product user manual for more information about operating modes and measurement intervals.

Received uplink messages from the device will appear under the APPLICATIONS DATA field. These are indicated by an *upwards facing blue arrow*. By default, incoming data packets are received in binary format (e.g 13 6A 00 00 00).

APPLICATION DATA || pause 🗑️ clear

Filters: uplink downlink activation ack error

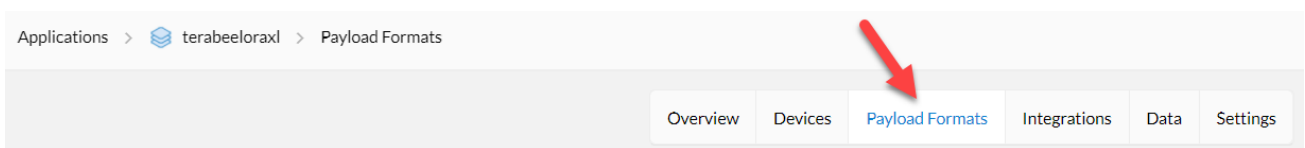
	time	counter	port	
▲	14:43:11	8	6	payload: 1F 8B 00 00 00
▲	14:40:45	7	6	payload: 1F 7C 00 00 00
▲	14:38:18	6	6	payload: 1F 8B 00 00 00
▲	14:35:51	5	6	payload: 1F 8B 00 00 00
▲	14:33:24	4	6	payload: 1F 86 00 00 00
▲	14:30:57	3	6	payload: 1F 8B 00 00 00
▲	14:28:30	2	6	payload: 13 6A 00 00 00
▲	14:26:02	1	6	retry payload: 13 6A 00 00 00
⚡	14:25:36	dev addr: 26 01 4C 30 app eui: 70 B3 D5 7E D0 03 FF C1 dev eui: 00 BFA 4 88 81 BAE 9 88		

Downlink messages - from the gateway to the Terabee device - are represented by a downwards facing (blue) arrow in the DATA dialog window. These will appear in case the UPLINK CONFIRMATION' feature is ENABLED on the Terabee device.

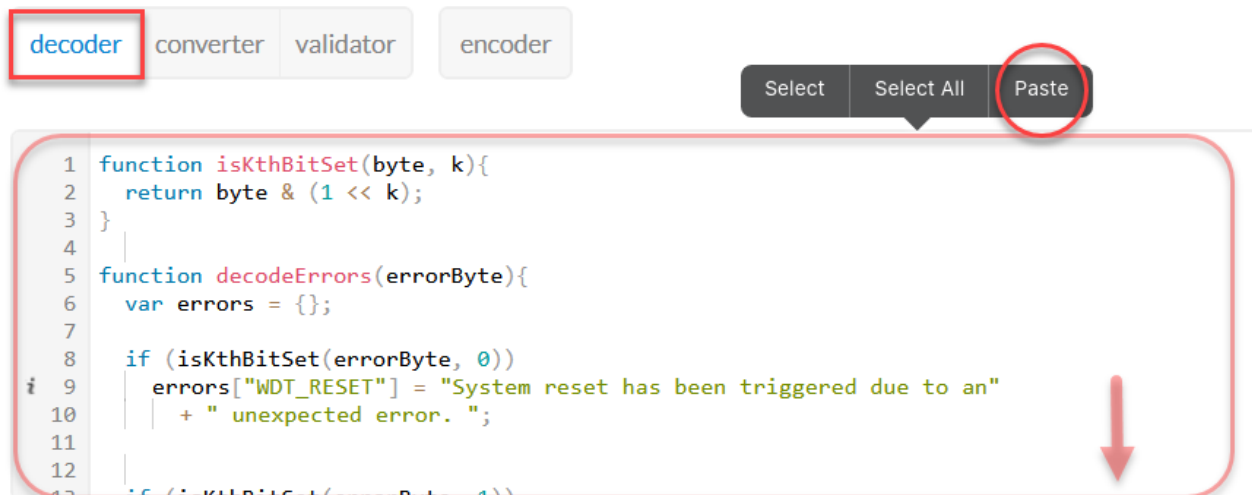
Decoding uplink payload

Data packets from the Terabee device to the network server are communicated in an encrypted format (binary). In order to decrypt these packets into objects (readable by human), the user needs to apply a decoding function.

1. Under the created application, select PAYLOAD FORMATS section in the upper right menu.

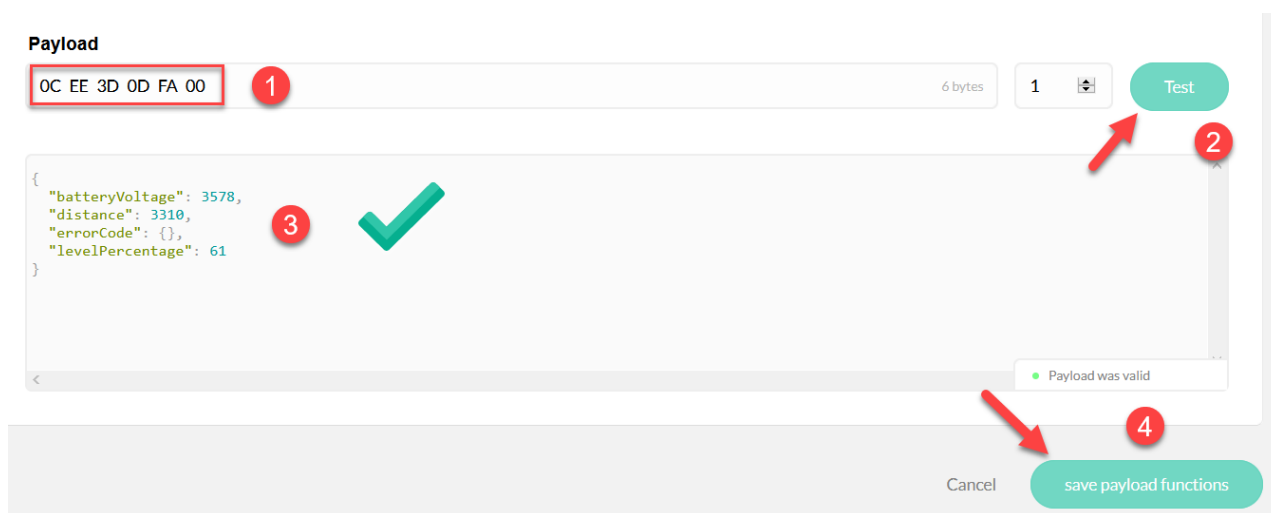


2. Copy the provided decoding function (Javascript) into the DECODER area. Please consult **Appendix A** for more information about the payload structure and decoding function.



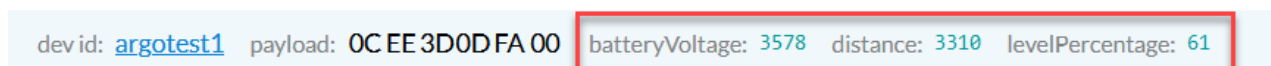
3. Before saving the decoding function, the user has the possibility to test it. To do so, copy one of the uplink data payloads previously received from the DATA field and paste it into the PAYLOAD TEST field.

Click on TEST. An example of decoded payload - received from the Terabee device - should now be available in the payload dialog window.



4. Click on SAVE PAYLOAD FUNCTIONS (bottom left side of the window) to apply and register the newly added decoding function.

For every new uplink message received, the DATA dialog box will now also display the decoded information (objects) in addition to the binary payload.



Appendix A

General uplink payload structure

Distance data	Remaining material level	Battery voltage	Error codes
byte [0] - MSB byte [1] - LSB	byte [2]	byte [3] - MSB byte [4] - LSB	byte [5]

MSB = most significant byte, LSB = least significant byte

Error codes (byte 5) bit structure

BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0
RFU	RFU	RFU	RFU	TOF-TME OUT	BATTERY	LORA_LO W_POWE R_FAIL	WDT_RES ET

RFU = reserved for future use

Uplink payload decoding function

```
function isKthBitSet(byte, k){
  return byte & (1 << k);
}

function decodeErrors(errorByte){
  var errors = {};

  if (isKthBitSet(errorByte, 0))
    errors["WDT_RESET"] = "System reset has been triggered due to an"
      + " unexpected error. ";

  if (isKthBitSet(errorByte, 1))
    errors["LORA_LOW_POWER_FAIL"] = "Embedded LoRa module failed to"
      + " enter sleep mode. If possible, please restart the device. ";
}
```



```

if (isKthBitSet(errorByte, 2))
  errors["BATTERY"] = "System failed to read voltage from the battery."
  + " If possible, please restart the device.";

if (isKthBitSet(errorByte, 3))
  errors["TOF_TIMEOUT"] = "System failed to get a response from the"
  + " onboard ToF distance sensor. If possible, please restart the device.";
return errors;
}

function decodeLevelPercentage(level){
  if (level === 255)
    return "LEVEL_ERROR";
  return level;
}

function decodeDistance(distance){
  if (distance === 0)
    return "TARGET_TOO_CLOSE";

  if (distance === 65535)
    return "TARGET_TOO_FAR";

  if (distance === 1)
    return "INVALID_READING";
  return distance;
}

function Decoder(bytes, port) {
  var distance = decodeDistance(bytes[0]<<8 | bytes[1]);
  var levelPercentage = decodeLevelPercentage(bytes[2]);
  var batteryVoltage = bytes[3]<<8 | bytes[4];
  var errorCode = decodeErrors(bytes[5]);

  return {
    distance: distance,
    levelPercentage: levelPercentage,
    batteryVoltage: batteryVoltage,
    errorCode: errorCode
  };
}

```